

Laboratório 05:

Gerador de Código Intermediário Linguagem COMP-ITA 2017

Alunos: Gabriel Brito e Lucas França de Oliveira

Professor: Fábio C. Mokarzel

Disciplina: CES-41

20 de Maio de 2017

**Introdução**

A atividade realizada consistiu em implementar o gerador de código intermediário da linguagem COMP-ITA 2017*.* A fim de demonstrar o seu bom funcionamento, será fornecido como entrada vários programas escritos na referida linguagem e com as respectivas listas de quádruplas geradas.

**Desenvolvimento**

Para a realização da atividade, foram adaptados os arquivos da atividade de laboratório 4: “build.l” e “build.y”.

**Analisador Léxico**

O analisador léxico encontra-se no arquivo “lab5.l”. O conteúdo do arquivo é:

delim [ \t\n\r]

ws {delim}+

digito [0-9]

letra [A-Za-z]

intct {digito}+

floatct {digito}+((\.{digito}\*)|((\.{digito}\*)?([Ee][+-]?{digito}+)))

id {letra}({letra}|{digito})\*

carac1 \\.|[^\\']

charct '{carac1}'

carac2 \\.|[^\\\"]

string \"{carac2}\*\"

%%

{ws} { ;}

"/\*" {comentario();}

call {return CALL;}

char {return CHAR;}

else {return ELSE;}

false {return FALSE;}

float {return FLOAT;}

for {return FOR;}

function {return FUNCTION;}

if {return IF;}

int {return INT;}

logic {return LOGIC;}

procedure {return PROCEDURE;}

program {return PROGRAM;}

read {return READ;}

repeat {return REPEAT;}

return {return RETURN;}

true {return TRUE;}

var {return VAR;}

while {return WHILE;}

write {return WRITE;}

{id} {strcpy (yylval.string, yytext); return ID;}

{floatct} {yylval.valreal = atof(yytext); return FLOATCT;}

{intct} {yylval.valint = atoi(yytext); return INTCT;}

{charct} {strcpy (yylval.string, yytext+1); yylval.string[strlen(yylval.string)-1]='\0'; return CHARCT;}

{string} {strcpy (yylval.string, yytext+1); yylval.string[strlen(yylval.string)-1]='\0'; return STRING;}

"||" {return OR;}

"&&" {return AND;}

"!" {return NOT;}

"<" {yylval.atr = LT; return RELOP;}

"<=" {yylval.atr = LE; return RELOP;}

">" {yylval.atr = GT; return RELOP;}

">=" {yylval.atr = GE; return RELOP;}

"==" {yylval.atr = EQ; return RELOP;}

"!=" {yylval.atr = NE; return RELOP;}

"+" {yylval.atr = PL; return ADOP;}

"-" {yylval.atr = MI; return ADOP;}

"\*" {yylval.atr = MU; return MULTOP;}

"/" {yylval.atr = DI; return MULTOP;}

"%" {yylval.atr = RE; return MULTOP;}

"~" {return NEG;}

"(" {return OPPAR;}

")" {return CLPAR;}

"[" {return OPBRAK;}

"]" {return CLBRAK;}

"{" {return OPBRACE;}

"}" {return CLBRACE;}

";" {return SCOLON;}

"," {return COMMA;}

"=" {return ASSIGN;}

. {yylval.carac = yytext[0]; return INVAL;}

%%

void comentario()

{

char c;

int estado = 1;

while (estado != 3) {

switch (estado) {

case 1:

c = input ();

if (c == EOF) estado = 3;

else if (c == '\*') estado = 2;

break;

case 2:

c = input ();

if (c == EOF) estado = 3;

else if (c == '/') estado = 3;

else if (c == '\*') estado = 2;

else estado = 1;

break;

default:

break;

}

}

}

char tratachar(char \*s)

{

if (s[1] != '\\') return s[1];

else switch(s[2]) {

case 'a': return 7;

case 'b': return 8;

case '\"': return 34;

case 't': return 9;

case '0': return 0;

case 'v': return 11;

case '\\': return 92;

case 'r': return 13;

case 'f': return 12;

case 'n': return 10;

case '\'': return 39;

default: return s[2];

}

return s[2];

}

void tabular()

{

int i;

for(i=0; i<tab; i++) printf(" ");

}

**Analisador Sintático**

O analisador sintático encontra-se no arquivo “lab5.y”. O conteúdo do arquivo é:

[*Copiar o conteúdo do arquivo YACC*]

**Testes**

A fim de demonstrar o bom funcionamento do gerador de código intermediário, foram realizados vários testes com diferentes programas escrito na linguagem COMP-ITA 2017. Os testes encontram-se a seguir:

**1. Programa “teste” do arquivo “inter012017.dat”**

Entrada:

program teste;

var

int i, j, m;

logic b1, b2, b3;

float k;

{

i = ~4;

j = ~i\*3;

k = i / j;

m = i % j;

b2 = true;

b3 = !b2;

b1 = (i+3\*j >= j/2-2) && !b2 || (b1 != b2) && (i == j);

}

Saída:

1) OPENMOD, (MODULO, teste), (IDLE), (IDLE)

2) MENUN, (INT, 4), (IDLE), (VAR, ##1)

3) ATRIB, (VAR, ##1), (IDLE), (VAR, i)

4) MENUN, (VAR, i), (IDLE), (VAR, ##2)

5) MULT, (VAR, ##2), (INT, 3), (VAR, ##3)

6) ATRIB, (VAR, ##3), (IDLE), (VAR, j)

7) DIV, (VAR, i), (VAR, j), (VAR, ##4)

8) ATRIB, (VAR, ##4), (IDLE), (VAR, k)

9) RESTO, (VAR, i), (VAR, j), (VAR, ##5)

10) ATRIB, (VAR, ##5), (IDLE), (VAR, m)

11) ATRIB, (LOGIC, 1), (IDLE), (VAR, b2)

12) NOT, (VAR, b2), (IDLE), (VAR, ##6)

13) ATRIB, (VAR, ##6), (IDLE), (VAR, b3)

14) MULT, (INT, 3), (VAR, j), (VAR, ##7)

15) MAIS, (VAR, i), (VAR, ##7), (VAR, ##8)

16) DIV, (VAR, j), (INT, 2), (VAR, ##9)

17) MENOS, (VAR, ##9), (INT, 2), (VAR, ##10)

18) GE, (VAR, ##8), (VAR, ##10), (VAR, ##11)

19) NOT, (VAR, b2), (IDLE), (VAR, ##12)

20) AND, (VAR, ##11), (VAR, ##12), (VAR, ##13)

21) NE, (VAR, b1), (VAR, b2), (VAR, ##14)

22) EQ, (VAR, i), (VAR, j), (VAR, ##15)

23) AND, (VAR, ##14), (VAR, ##15), (VAR, ##16)

24) OR, (VAR, ##13), (VAR, ##16), (VAR, ##17)

25) ATRIB, (VAR, ##17), (IDLE), (VAR, b1)

**2. Teste para comando IF sem ELSE**

Entrada:

program teste;

var

int i, j, n, v, h, a;

float b, c;

{

i = 4;

j = 3;

v = i / j;

h = i % j;

if(i < j)

{

i = i + 3;

n = v / h;

}

i = 0;

}

Saída:

Quadruplas do modulo teste:

1) OPENMOD, (MODULO, teste), (IDLE), (IDLE)

2) ATRIB, (INT, 4), (IDLE), (VAR, i)

3) ATRIB, (INT, 3), (IDLE), (VAR, j)

4) DIV, (VAR, i), (VAR, j), (VAR, ##1)

5) ATRIB, (VAR, ##1), (IDLE), (VAR, v)

6) RESTO, (VAR, i), (VAR, j), (VAR, ##2)

7) ATRIB, (VAR, ##2), (IDLE), (VAR, h)

8) LT, (VAR, i), (VAR, j), (VAR, ##3)

9) JF, (VAR, ##3), (IDLE), (ROTULO, 14)

10) MAIS, (VAR, i), (INT, 3), (VAR, ##4)

11) ATRIB, (VAR, ##4), (IDLE), (VAR, i)

12) DIV, (VAR, v), (VAR, h), (VAR, ##5)

13) ATRIB, (VAR, ##5), (IDLE), (VAR, n)

14) NOP, (IDLE), (IDLE), (IDLE)

15) ATRIB, (INT, 0), (IDLE), (VAR, i)

**3. Teste para IF com ELSE**

Entrada:

var

int i, j, n, v, h, a;

float b, c;

{

i = 4;

j = 3;

v = i / j;

h = i % j;

if(i < j)

{

i = i + 3;

n = v / h;

}

else

{

v = h;

i = j;

}

n = v;

i = 0;

}

Saída:

Quadruplas do modulo teste:

1) OPENMOD, (MODULO, teste), (IDLE), (IDLE)

2) ATRIB, (INT, 4), (IDLE), (VAR, i)

3) ATRIB, (INT, 3), (IDLE), (VAR, j)

4) DIV, (VAR, i), (VAR, j), (VAR, ##1)

5) ATRIB, (VAR, ##1), (IDLE), (VAR, v)

6) RESTO, (VAR, i), (VAR, j), (VAR, ##2)

7) ATRIB, (VAR, ##2), (IDLE), (VAR, h)

8) LT, (VAR, i), (VAR, j), (VAR, ##3)

9) JF, (VAR, ##3), (IDLE), (ROTULO, 15)

10) MAIS, (VAR, i), (INT, 3), (VAR, ##4)

11) ATRIB, (VAR, ##4), (IDLE), (VAR, i)

12) DIV, (VAR, v), (VAR, h), (VAR, ##5)

13) ATRIB, (VAR, ##5), (IDLE), (VAR, n)

14) JUMP, (IDLE), (IDLE), (ROTULO, 18)

15) NOP, (IDLE), (IDLE), (IDLE)

16) ATRIB, (VAR, h), (IDLE), (VAR, v)

17) ATRIB, (VAR, j), (IDLE), (VAR, i)

18) NOP, (IDLE), (IDLE), (IDLE)

19) ATRIB, (VAR, v), (IDLE), (VAR, n)

20) ATRIB, (INT, 0), (IDLE), (VAR, i)

**4. Teste para commando WHILE:**

Entrada:

program teste;

var

int i, j, n, v, h, a;

float b, c;

{

i = 4;

j = 3;

v = i / j;

h = i % j;

while(i < j)

i = j + h;

i = 0;

}

Saída:

Quadruplas do modulo teste:

1) OPENMOD, (MODULO, teste), (IDLE), (IDLE)

2) ATRIB, (INT, 4), (IDLE), (VAR, i)

3) ATRIB, (INT, 3), (IDLE), (VAR, j)

4) DIV, (VAR, i), (VAR, j), (VAR, ##1)

5) ATRIB, (VAR, ##1), (IDLE), (VAR, v)

6) RESTO, (VAR, i), (VAR, j), (VAR, ##2)

7) ATRIB, (VAR, ##2), (IDLE), (VAR, h)

8) NOP, (IDLE), (IDLE), (IDLE)

9) LT, (VAR, i), (VAR, j), (VAR, ##3)

10) JF, (VAR, ##3), (IDLE), (ROTULO, 14)

11) MAIS, (VAR, j), (VAR, h), (VAR, ##4)

12) ATRIB, (VAR, ##4), (IDLE), (VAR, i)

13) JUMP, (IDLE), (IDLE), (ROTULO, 8)

14) NOP, (IDLE), (IDLE), (IDLE)

15) ATRIB, (INT, 0), (IDLE), (VAR, i)

**5. Teste para comando READ e WRITE com variáveis escalares:**

Entrada:

program teste;

var

int i, j, n, v, h, a;

float b, c;

{

i = 4;

j = 3;

v = i / j;

h = i % j;

read(a, b, c);

write("Valor de a+b:", a + b, "Valor de c:", c);

i = 0;

}

Saida:

Quadruplas do modulo teste:

1) OPENMOD, (MODULO, teste), (IDLE), (IDLE)

2) ATRIB, (INT, 4), (IDLE), (VAR, i)

3) ATRIB, (INT, 3), (IDLE), (VAR, j)

4) DIV, (VAR, i), (VAR, j), (VAR, ##1)

5) ATRIB, (VAR, ##1), (IDLE), (VAR, v)

6) RESTO, (VAR, i), (VAR, j), (VAR, ##2)

7) ATRIB, (VAR, ##2), (IDLE), (VAR, h)

8) PARAM, (VAR, a), (IDLE), (IDLE)

9) PARAM, (VAR, b), (IDLE), (IDLE)

10) PARAM, (VAR, c), (IDLE), (IDLE)

11) READ, (INT, 3), (IDLE), (IDLE)

12) PARAM, (CADEIA, Valor de a+b:), (IDLE), (IDLE)

13) MAIS, (VAR, a), (VAR, b), (VAR, ##3)

14) PARAM, (VAR, ##3), (IDLE), (IDLE)

15) PARAM, (CADEIA, Valor de c:), (IDLE), (IDLE)

16) PARAM, (VAR, c), (IDLE), (IDLE)

17) WRITE, (INT, 4), (IDLE), (IDLE)

18) ATRIB, (INT, 0), (IDLE), (VAR, i)

**6. Teste para comando REPEAT WHILE:**

Entrada:

program teste;

var

int i, j, n, v, h, a;

float b, c;

{

i = 4;

j = 3;

v = i / j;

h = i % j;

repeat

i = i + 1;

while(i < 5);

i = 0;

}

Saída:

Quadruplas do modulo teste:

1) OPENMOD, (MODULO, teste), (IDLE), (IDLE)

2) ATRIB, (INT, 4), (IDLE), (VAR, i)

3) ATRIB, (INT, 3), (IDLE), (VAR, j)

4) DIV, (VAR, i), (VAR, j), (VAR, ##1)

5) ATRIB, (VAR, ##1), (IDLE), (VAR, v)

6) RESTO, (VAR, i), (VAR, j), (VAR, ##2)

7) ATRIB, (VAR, ##2), (IDLE), (VAR, h)

8) NOP, (IDLE), (IDLE), (IDLE)

9) MAIS, (VAR, i), (INT, 1), (VAR, ##3)

10) ATRIB, (VAR, ##3), (IDLE), (VAR, i)

11) LT, (VAR, i), (INT, 5), (VAR, ##4)

12) JF, (VAR, ##4), (IDLE), (ROTULO, 14)

13) JUMP, (IDLE), (IDLE), (ROTULO, 8)

14) NOP, (IDLE), (IDLE), (IDLE)

15) ATRIB, (INT, 0), (IDLE), (VAR, i)

**7. Teste para comando FOR:**

Entrada:

program teste;

var

int i, j, n, v, h, a;

float b, c;

{

i = 4;

j = 3;

v = i / j;

h = i % j;

for(i = 1; i <= 50; i = i + 1)

{

write("-");

h = h % 10;

}

i = 0;

}

Saída:

Quadruplas do modulo teste:

1) OPENMOD, (MODULO, teste), (IDLE), (IDLE)

2) ATRIB, (INT, 4), (IDLE), (VAR, i)

3) ATRIB, (INT, 3), (IDLE), (VAR, j)

4) DIV, (VAR, i), (VAR, j), (VAR, ##1)

5) ATRIB, (VAR, ##1), (IDLE), (VAR, v)

6) RESTO, (VAR, i), (VAR, j), (VAR, ##2)

7) ATRIB, (VAR, ##2), (IDLE), (VAR, h)

8) ATRIB, (INT, 1), (IDLE), (VAR, i)

9) NOP, (IDLE), (IDLE), (IDLE)

10) LE, (VAR, i), (INT, 50), (VAR, ##3)

11) JF, (VAR, ##3), (IDLE), (ROTULO, 19)

12) PARAM, (CADEIA, -), (IDLE), (IDLE)

13) WRITE, (INT, 1), (IDLE), (IDLE)

14) RESTO, (VAR, h), (INT, 10), (VAR, ##5)

15) ATRIB, (VAR, ##5), (IDLE), (VAR, h)

16) MAIS, (VAR, i), (INT, 1), (VAR, ##4)

17) ATRIB, (VAR, ##4), (IDLE), (VAR, i)

18) JUMP, (IDLE), (IDLE), (ROTULO, 9)

19) NOP, (IDLE), (IDLE), (IDLE)

20) ATRIB, (INT, 0), (IDLE), (VAR, i)

**8. Teste para commandos utilizando variáveis indexadas:**

Entrada:

program teste;

var

int i, j, k, A[10][10];

{

i = 7;

j = 5;

k = A[i - 3][j + 2] + 5;

A[10 - i][9 - j] = i + j \* k;

read(i, j, A[2][3], k);

read(A[1][2]);

}

Saída:

Quadruplas do modulo teste:

1) OPENMOD, (MODULO, teste), (IDLE), (IDLE)

2) ATRIB, (INT, 7), (IDLE), (VAR, i)

3) ATRIB, (INT, 5), (IDLE), (VAR, j)

4) MENOS, (VAR, i), (INT, 3), (VAR, ##1)

5) IND, (VAR, ##1), (IDLE), (IDLE)

6) MAIS, (VAR, j), (INT, 2), (VAR, ##2)

7) IND, (VAR, ##2), (IDLE), (IDLE)

8) INDEX, (VAR, A), (INT, 2), (VAR, ##3)

9) CONTAPONT, (VAR, ##3), (IDLE), (VAR, ##4)

10) MAIS, (VAR, ##3), (INT, 5), (VAR, ##5)

11) ATRIB, (VAR, ##5), (IDLE), (VAR, k)

12) MENOS, (INT, 10), (VAR, i), (VAR, ##6)

13) IND, (VAR, ##6), (IDLE), (IDLE)

14) MENOS, (INT, 9), (VAR, j), (VAR, ##7)

15) IND, (VAR, ##7), (IDLE), (IDLE)

16) INDEX, (VAR, A), (INT, 2), (VAR, ##8)

17) MULT, (VAR, j), (VAR, k), (VAR, ##9)

18) MAIS, (VAR, i), (VAR, ##9), (VAR, ##10)

19) ATRIBPONT, (VAR, ##10), (IDLE), (VAR, ##8)

20) PARAM, (VAR, i), (IDLE), (IDLE)

21) PARAM, (VAR, j), (IDLE), (IDLE)

22) READ, (INT, 2), (IDLE), (IDLE)

23) IND, (INT, 2), (IDLE), (IDLE)

24) IND, (INT, 3), (IDLE), (IDLE)

25) INDEX, (VAR, A), (INT, 2), (VAR, ##11)

26) PARAM, (VAR, ##12), (IDLE), (IDLE)

27) READ, (INT, 1), (IDLE), (IDLE)

28) ATRIBPONT, (VAR, ##12), (IDLE), (VAR, ##11)

29) PARAM, (VAR, k), (IDLE), (IDLE)

30) READ, (INT, 1), (IDLE), (IDLE)

31) IND, (INT, 1), (IDLE), (IDLE)

32) IND, (INT, 2), (IDLE), (IDLE)

33) INDEX, (VAR, A), (INT, 2), (VAR, ##13)

34) PARAM, (VAR, ##14), (IDLE), (IDLE)

35) READ, (INT, 1), (IDLE), (IDLE)

36) ATRIBPONT, (VAR, ##14), (IDLE), (VAR, ##13)

**9. Teste utilizando o programa “primos” com vários casos de uso das funções da linguagem:**

Entrada:

program primos;

var

logic achou; int n, div, resto, cont, num;

{

read (n);

write (n, " primeiros numeros naturais primos:\n\n");

if (n > 0) {

num = 1; cont = 0;

while (cont < n) {

num = num + 1; div = 2; achou = false;

while (achou == false && div \* div <= num) {

resto = num % div;

if (resto == 0) achou = true;

else div = div + 1;

}

if (achou == false) {

write (num, "\n"); cont = cont + 1;

}

}

}

}

Saída:

Quadruplas do modulo primos:

1) OPENMOD, (MODULO, primos), (IDLE), (IDLE)

2) PARAM, (VAR, n), (IDLE), (IDLE)

3) READ, (INT, 1), (IDLE), (IDLE)

4) PARAM, (VAR, n), (IDLE), (IDLE)

5) PARAM, (CADEIA, primeiros numeros naturais primos:\n\n), (IDLE), (IDLE)

6) WRITE, (INT, 2), (IDLE), (IDLE)

7) GT, (VAR, n), (INT, 0), (VAR, ##1)

8) JF, (VAR, ##1), (IDLE), (ROTULO, 46)

9) ATRIB, (INT, 1), (IDLE), (VAR, num)

10) ATRIB, (INT, 0), (IDLE), (VAR, cont)

11) NOP, (IDLE), (IDLE), (IDLE)

12) LT, (VAR, cont), (VAR, n), (VAR, ##2)

13) JF, (VAR, ##2), (IDLE), (ROTULO, 45)

14) MAIS, (VAR, num), (INT, 1), (VAR, ##3)

15) ATRIB, (VAR, ##3), (IDLE), (VAR, num)

16) ATRIB, (INT, 2), (IDLE), (VAR, div)

17) ATRIB, (LOGIC, 0), (IDLE), (VAR, achou)

18) NOP, (IDLE), (IDLE), (IDLE)

19) EQ, (VAR, achou), (LOGIC, 0), (VAR, ##4)

20) MULT, (VAR, div), (VAR, div), (VAR, ##5)

21) LE, (VAR, ##5), (VAR, num), (VAR, ##6)

22) AND, (VAR, ##4), (VAR, ##6), (VAR, ##7)

23) JF, (VAR, ##7), (IDLE), (ROTULO, 35)

24) RESTO, (VAR, num), (VAR, div), (VAR, ##8)

25) ATRIB, (VAR, ##8), (IDLE), (VAR, resto)

26) EQ, (VAR, resto), (INT, 0), (VAR, ##9)

27) JF, (VAR, ##9), (IDLE), (ROTULO, 30)

28) ATRIB, (LOGIC, 1), (IDLE), (VAR, achou)

29) JUMP, (IDLE), (IDLE), (ROTULO, 33)

30) NOP, (IDLE), (IDLE), (IDLE)

31) MAIS, (VAR, div), (INT, 1), (VAR, ##10)

32) ATRIB, (VAR, ##10), (IDLE), (VAR, div)

33) NOP, (IDLE), (IDLE), (IDLE)

34) JUMP, (IDLE), (IDLE), (ROTULO, 18)

35) NOP, (IDLE), (IDLE), (IDLE)

36) EQ, (VAR, achou), (LOGIC, 0), (VAR, ##11)

37) JF, (VAR, ##11), (IDLE), (ROTULO, 43)

38) PARAM, (VAR, num), (IDLE), (IDLE)

39) PARAM, (CADEIA, \n), (IDLE), (IDLE)

40) WRITE, (INT, 2), (IDLE), (IDLE)

41) MAIS, (VAR, cont), (INT, 1), (VAR, ##12)

42) ATRIB, (VAR, ##12), (IDLE), (VAR, cont)

43) NOP, (IDLE), (IDLE), (IDLE)

44) JUMP, (IDLE), (IDLE), (ROTULO, 11)

45) NOP, (IDLE), (IDLE), (IDLE)

46) NOP, (IDLE), (IDLE), (IDLE)